Detection and Prognostics on Low Dimensional Systems

Ashok N. Srivastava, *Member, IEEE,* Santanu Das

The final draft of this work was accepted for publication on August 29, 2008 in the IEEE Transactions on Systems Man and Cybernetics, Part C

***Abstract*—This paper describes the application of known and novel prognostic algorithms on systems that can be described by low dimensional, potentially nonlinear dynamics. The methods rely on estimating the conditional probability distribution of the output of the system at a future time given knowledge of the current state of the system. We show how to estimate these conditional probabilities using a variety of techniques, including bagged neural networks and kernel methods such as Gaussian Process Regression (GPR). The results are compared with standard methods such as a linear autoregressive model and the nearest neighbor algorithm. We demonstrate the algorithms on a real-world data set and a simulated data set. The real-world data set consists of the intensity of an NH_3 laser. The laser data set has been shown by other authors to exhibit low-dimensional chaos with significant drops in intensity. The simulated data set is generated from the Lorenz attractor and has known statistical characteristics. On these data sets, we show the evolution of the estimated conditional probability distribution, the way it can act as a prognostic signal, and its use as an early warning system. We also review a novel approach to perform Gaussian Process Regression with large numbers of data points.**

I. Introduction

This paper addresses the problem of making predictions of future events on systems that can be described by low-dimensional dynamical equations. We assume that we are given data from a data generating process that can be functionally described by the following equations:

$$\mathbf{h}_t = \Gamma(\mathbf{h}^*_{t-1}) \quad (1)$$

$$\mathbf{x}_t = \Psi(\mathbf{x}^*_{t-1}, \mathbf{h}^*_t, u_t) \quad (2)$$

$$y_t = \Omega(\mathbf{x}_t) \quad (3)$$

We assume that the function Γ determining the evolution of the hidden system state $\mathbf{h}_t$ is unknown. We also assume that the function Ψ, which generates the observed output of the system is unknown. We assume that the vector $\mathbf{x}$ is an N dimensional state vector, and $\mathbf{x}^*_{t-1}$ is its history for the last D time steps: $\mathbf{x}^*_{t-1} = [\mathbf{x}_{t-D}, \mathbf{x}_{t-D+1}, ..., \mathbf{x}_{t-1}]$. The quantity u_t is the observed system input, and y_t is the observed scalar system output. We assume that the entire data that is available, covering both inputs and outputs is given by the set $(\mathcal{X}, \mathcal{Y})$.

The hidden state $\mathbf{h}_t$ is assumed to correspond to different mode configurations within the system. In the case where we assume that the hidden state takes on discrete values, $\mathbf{h}_t$ switches between M modes, each affecting the output dynamics Ψ. In the case of a failure of the system, $\mathbf{h}_t$ could move to a failed state, thus also changing the nature of the observed output. In other applications, $\mathbf{h}_t$ could be a continuous state variable, modeling a slow progression of the system from a normal state to a failed state [27]. Because we assume that we do not know the output function Ψ or the hidden state $\mathbf{h}_t$, we cannot rely on it to help us determine whether or not the observed y_t is anomalous. The problem that we address in this paper is to develop a method to discover whether or not the current observed value y_t represents an anomaly based on the observed history of the system.

Several approaches have been discussed in the literature to address the problem of making future predictions on systems that can be described by Equations 1, 2 and 3. Traditional approaches include those developed in the system identification community [16]. Other techniques include Hidden Markov Models, where the transitions between the M hidden discrete states are modeled as a first-order hidden Markov process [22]. The HMM allows for the dynamics of the system to be modeled but requires that a procedure (such as clustering or learning vector quantization) be used to develop a discrete representation of the system output. Other popular methods to convert the time series into symbolic representations include Piecewise Aggregate Approximation (PAA) [17] and Symbolic Aggregate approXimation (SAX) [18]. Once the symbolic representation is generated it can be analyzed using the HMM. For many applications, the dwell time within a hidden state does not follow the exponential decay that arises from the standard HMM

algorithm. Dong and He [7] [8] have recently developed this method for analyzing systems with hidden discrete transitions (as shown in Figure 1) using a hidden semi-Markov model (HSMM) where the dwell time within a state is modeled by a Gaussian distribution. Their work shows that the HSMM can lead to superior performance on real-world applications compared with the standard HMM formulation.

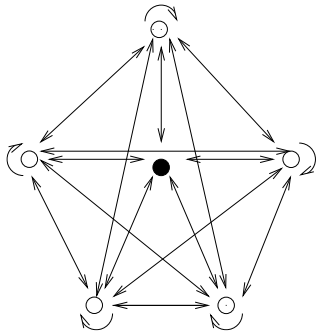

Fig. 1. This figure shows a finite state machine of the system states. The finite state machine has a potential path for progression from normal operation (clear circles) to the failed state (solid circle). The model allows for the system to move from a failed state back to normal operation which models intermittent problems that can arise in complex systems. For generality, we have included bi-directional arrows and a fully connected graph.

II. Motivation

We demonstrate the detection and prognostic capabilities of our algorithm on the data from the Lorenz model and the NH_3 laser system because it is a suitable test bed for building prognostic algorithms for several reasons including:

- The NH_3 laser system can be modeled by low-dimensional differential equations, thus allowing for the application of Takens' Theorem from nonlinear dynamics. This theorem gives the mathematical foundation for us to take past values of the time series to predict future values.

- The laser intensity can be approximated by the Lorenz equations which are known to exhibit chaotic behavior for certain values of the parameters which puts a bound on the long-term predictability of the intensity. Chaos theory tells us that short-term predictions may be possible for chaotic time series, but long-term predictions are *impossible*. The time horizon for such predictions is given by the Lyapunov time constant of the system, which can be derived from data or directly from the governing equations of a system [6].

- The NH_3 laser data has been widely used in the time series prediction community for over a decade to test machine learning algorithms for detection, prediction, and model validation [15], [39].

Methods that are suitable for making predictions on this system may be transferable to other systems. The capability to predict time series that arise from nonlinear chaotic systems could be useful in several other real-world applications that exhibit chaotic behavior. Fault detection systems make use of either passive or active sensing devices, connected in a discrete or continuous fashion [5], [29], [34], [35].

These sensing devices monitor one or more state variables depending on the nature of the application. For some systems, these measured state variables can be analogous to state variables in differential equations. For example, Yamanaka et. al. [1], [36], [37] presented a simple analytical model to explain the interaction between crack planes using van der Waals inter-atomic force. This approach addressed the problem of detecting a closed crack in a mechanical system using ultrasonic testing. The paper proposed a new detection technique based on an analysis of the subharmonic components which are generated due to the nonlinear interaction of the crack surface and the forcing function. The authors showed that under certain parametric conditions the vibration signals that represent the crack opening displacement can exhibit chaotic behavior.

Foong et al. [10] conducted a separate study to shown that the response of nonlinear dynamical systems can have chaotic oscillation under fatigue crack growth. Most physical phenomenon are complex and high dimensional in nature which can make the modeling process difficult. In some cases, it is possible to approximate the high-dimensional system with low-dimensional dynamics which are informative and interpretable.

III. Background

A time series is a collection of observations represented sequentially as a function of time. A significant body of work is to create mathematical models that predict the system behavior from a set of observations. The approach taken in the machine-learning community has been to create potentially nonlinear statistical models that learn a mapping from past states to future states without requiring extensive knowledge of the physical system. In many cases, methods such as neural networks can produce high quality predictions [32].

The focus of the current study is on the analysis of time series obtained from a low-dimensional nonlinear dynamical system that exhibits chaotic behavior. The objective is to develop a prognostic algorithm that models the system dynamics using past observations and forecasts the future system behavior while providing a measure of uncertainty in the predictions. This measure of uncertainty can be interpreted as the model's confidence in the prediction. If the model's confidence reduces at a certain point, it could be indicative of an unexpected event, thus leading to a prognostic signal.

Because the system is chaotic, there is an upper bound on the forecasting horizon [32]. Once the model is able to make predictions of the system state at time $t+1$ it can be iterated to make predictions for future time steps. This is done by updating the model with the current estimate of the output at each iteration and repeating the predictor for k number of steps in the future to generate estimates for the system output at times $t+1, t+2, ..., t+k$.

The idea to predict the future values of a time series as a linear combination of the preceding values was first introduced by Yule [38] as the auto-regressive process (AR). A comprehensive literature review on conventional techniques to select a model that could be used to forecast the behavior of the time has been provided in [32]. The authors provide insight on local-linear models, global autoregressive, moving average, and neural-networks based approaches such as the Radial Basis Function (RBF) based model. K-nearest neighbors, a local average model is determined by taking a weighted sum of outputs for the k inputs that are near the query vector. Using a different numbers of nearest neighbors, it is possible to find the optimal value of k for which the RMSE is minimized.

McNames [20] presented a new method of optimizing the model parameters in order to minimize the multi-step cross validation error. In previous work [20], the author proposed the adoption of nearest trajectory model for time-series prediction. This method searches the closest trajectory points in the reconstructed state space as opposed to nearest neighbors. A comparative study on different nonparametric methods including nearest neighbors, RBF, and nearest trajectory methods to predict chaotic time series can be found in [15]. The idea to obtain iterative time series predictions using a regression tree based approach has been addressed by Badel et. al. [2].

Gaussian Process models have gained popularity in the machine learning community because many machine learnign algorithms including neural networks, splines, and other regression methods are special cases of Gaussian Processes. These models are intended to predict the probability distribution of a future observation as characterized by the mean and variance of a Gaussian distribution. The variance acts as a measure of the uncertainty associated with each model prediction on future observations. A detailed review on various sources of uncertainties in modeling time series has been well documented by Draper [9].

In 1996, Neal [21] showed that Gaussian Process (GP) models are equivalent to the neural networks with one hidden layer with an infinite number of hidden neurons. Rasmussen [23] introduced the empirical formulation of Gaussian Processes in terms of probabilistic model using Bayesian treatment. Mackay [19] and Seeger [26] extended this research to show the relationship of the Gaussian Process model to several other popular machine learning techniques like generalized Radial Basis Functions (RBF), splines, and support vector machines.

Gaussian processes are fully specified by a mean function and covariance function. For zeros-mean process, the latter plays the prime role to characterize the process. The covariance function is equivalent to a Mercer kernel function that measures the similarity between two input points. In statistical terms, the kernel function calculates the covariance between the outputs corresponding to different inputs. The choice of the covariance function typically depends on the prior assumption on the smoothness and continuity of the underlying function generated by a process. Mathematically, a covariance function is valid if it produces a nonnegative definite covariance matrix for a given set of input points. Mackay [19] provides a detailed description on a wide variety of covariance functions.

IV. Main Idea

The main idea discussed in this paper is to build a predictive model that estimates y_t given the history of past observations. Rather than creating a single 'point-estimate' of y_t, we estimate $P(y_t|\mathbf{y}^*_{t-1})$. In this formulation, the mean of this quantity (obtained by computing the expected value) will produce an estimate for the future value of $\mathbf{y}_t$ while the variance of this distribution would quantify the uncertainty in the predictions. As that uncertainty changes in time, it can be indicative of an unanticipated change in the data generating process. This change could be due to several issues, including the movement of the hidden state $\mathbf{h}_t$ from one state to another.

For a true prognostic capability on low-dimensional systems, i.e., one where a forecast is made at a time horizon significantly far in the future compared to the

natural frequency of the system, we need the ability to make long term predictions. Such predictions are theoretically impossible for chaotic systems [6] if the prediction horizon is on the same order as the Lyapunov time. This quantity is the amount of time that is required for a volume of phase space to expand to a size that completely covers the underlying dynamic attractor.

There are at least two ways to make predictions about an event in the future. One method is to make the prediction in such a way that the estimate $P(y_{t+\tau}|y^*_{t-1})$ for a fixed duration τ in the future. If τ is significantly longer than the natural period of the system as measured by the low frequency modes in the Fourier spectrum of the signal, such a model would make predictions that are fixed in time; it would only make predictions for those events that are exactly τ units in the future. The second method is to generate *iterated predictions* which rely on developing a model to estimate $P(y_t|y^*_{t-1})$ and then feeding the output of the model back into the input, thus producing an estimate of $P(y_{t+1}|\mu_t, y^*_{t-1})$, where μ_t is a statistic (such as the mean) computed from the distribution computed at time t [25], [30]. This form of iterated prediction is depicted in Figure 2.

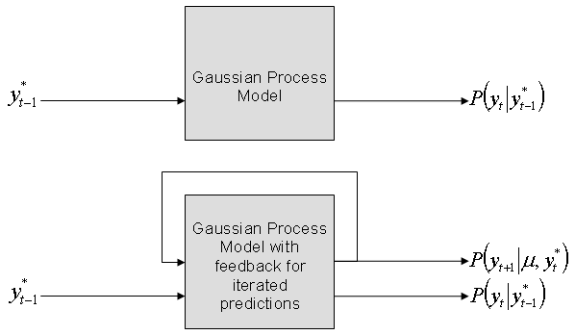

Fig. 2. The top panel in this figure shows a method for generating predictions of the quantity $P(y_t|\mathbf{y}^*_{t-1})$ using Gaussian Process regression. The lower panel shows a method to do iterated predictions, where a statistic such as the mean output of the model is fed back into the model to generate the next output.

V. Algorithms

We provide a brief overview of three algorithms that we use for estimating $P(y_t|\mathbf{y}^*_{t-1})$. The first two algorithms, the k-nearest neighbor and the bagged neural network algorithms, have been developed for many applications and have been widely discussed in the literature [13]. These algorithms provide a benchmark for comparison against the performance of the Gaussian Process regression which has become popular in the machine learning community over the last ten years.

Once these algorithms are appropriately trained, it is possible to iteratively predict q steps ahead in time for any given test case. It should be noted that for a chaotic time series the forecast of q steps in future is restricted to a prediction horizon that can be calculated from the Lyapunov exponent, if it is known. However to predict within this limit, it is necessary to train the model so that it learns the underlying dynamics of the system from the historical observations. The extraction of the dynamics can be achieved through delay coordinate embedding.

According to Taken's theorem [4], [28] given a finite set of scalar observations, it is possible to reconstruct the attractor in the phase space with an appropriate choice of time delay (τ) and embedding dimension (D). Given a time series x_t of length M the delay vectors z_t with length $M_r = M-(D-1)\tau$ data points can take the form as shown in Equation 4. Here the embedding dimension is an integer number and the delay is a duration with consecutive l sample points i.e. $\tau = l \times \frac{1}{f_s}$, where f_s is the frequency at which the data has been sampled. Given x_t, $\mathbf{X}$ serves as an input data matrix of size $(D-1)\times M_r$ and the i^{th} column of $\mathbf{X}$ represents a vector delayed by $(i-1)\tau$, where $i \leq (D-1)$. The corresponding output is denoted by $\mathcal{Y}$, termed as the target vector as shown in Equation 4.

A. K-Nearest Neighbor

The k-nearest neighbor algorithm uses all available input data $\mathbf{X}$ and associated output data $\mathcal{Y}$ until time t to produce a prediction of y_{t+1}. Specifically, to estimate $P(y_t|\mathbf{y}^*_{t-1})$, given $\mathbf{x}^*_t$ we identify the k vectors in the data set that are closest to that vector in terms of the Euclidean distance. The mean of the outputs associated with those k vectors is used as an estimate of the expected value of the distribution, $E_P(P(y_t|\mathbf{y}^*_{t-1}))$ and the variance of the outputs is used to estimate $Var_P(P(y_t|\mathbf{y}^*_{t-1}))$. This method does not attempt to summarize the data in any way and simply uses a simple 'look-up' table to estimate the parameters of the distribution. These estimates are based on heuristic principles regarding the local distribution of data at time t.

B. Bagged Neural Networks

The bagged neural network [3] model consists of developing N feedforward multi-layer perceptrons using the available input and output data. Each model is made by taking a sample of data (with replacement) from the data set $(\mathcal{X}, \mathcal{Y})$. If each network is labeled as $G_i(\theta_i)$,

$$z_t = \left[\overbrace{\begin{array}{cccc} x_{t_1} & x_{t_1+\tau} & \cdot\ \cdot & x_{t_1+(D-2)\tau} \\ x_{t_2} & x_{t_2+\tau} & \cdot\ \cdot & x_{t_2+(D-2)\tau} \\ x_{t_3} & x_{t_3+\tau} & \cdot\ \cdot & x_{t_3+(D-2)\tau} \\ \cdot & \cdot & \cdot\ \cdot & \cdot \\ \cdot & \cdot & \cdot\ \cdot & \cdot \\ x_{t_{M_r}} & x_{t_{M_r}+\tau} & \cdot\ \cdot & x_{t_{M_r}+(D-2)\tau} \end{array}}^{Input\ data\ matrix\ (\mathbf{X})} \overbrace{\begin{array}{c} x_{t_1+(D-1)\tau} \\ x_{t_2+(D-1)\tau} \\ x_{t_3+(D-1)\tau} \\ \cdot \\ \cdot \\ x_{t_{M_r}+(D-1)\tau} \end{array}}^{Target\ vector\ (\mathcal{Y})} \right] \quad (4)$$

the overall estimate of $E_P(P(y_t|\mathbf{y}^*_{t-1}))$ is given by:

$$E_P(P(y_t|\mathbf{y}^*_{t-1})) = \sum_{i=1}^{N} G_i(\mathcal{X}_i, \theta_i) \quad (5)$$

The estimate of the variance in the prediction, and thus the associated uncertainty is given by:

$$Var_P(P(y_t|\mathbf{y}^*_{t-1})) = \sum_{i=1}^{N} [G_i(\mathcal{X}_i, \theta_i) - E_P(P(y_t|\mathbf{y}^*_{t-1}))]^2 \quad (6)$$

This estimate of the uncertainty is a function of the heterogeneity of the data samples $\mathcal{X}_i$. In the event that all samples are identical, the only variation in the estimates will be due to variation in the initial starting point of the optimization procedure. This method is related to Ensemble Methods in machine learning and are widely used to estimate the mean and variance of the target distribution [31].

C. Gaussian Process Regression

A Gaussian Process is a stochastic process such that each finite subset of variables in the process is multivariate Gaussian distributed [24]. In 1996, Neal [21] noted that if the weights and biases in a neural network are drawn from a Gaussian distribution, then as the number of hidden units increases, the prior distribution over functions defined by such networks will converge to a Gaussian Process. This important result led many in the machine learning community to research Gaussian Processes and support vector machines.

Following the notation and derivation in Rassmussen and Williams [24], Gaussian Process Regression is a generalization of the standard linear regression model. We begin with a brief review of their Bayesian derivation of linear regression with the model $f(\mathbf{x}) = \mathbf{x}^T\mathbf{w}$ and an additive Gaussian noise, we have:

$$\begin{aligned} f(\mathbf{x}) &= \mathbf{x}^T\mathbf{w} \quad (7) \\ y &= f(\mathbf{x}) + \epsilon_n \\ \epsilon_n &\sim N(0, \sigma_n^2) \end{aligned}$$

where $\mathbf{x}$ is the test vector and $\mathbf{w}$ is a set of weights. Assuming that we choose a prior distribution for the weights as Gaussian with zero mean and covariance matrix Σ_p, and that the noise is Gaussian and independent and identically distributed, with variance σ_n^2, we compute the posterior probability distribution of the weights given the data $(\mathcal{X}, \mathbf{y})$:

$$P(\mathbf{w}|\mathbf{X}, \mathbf{y}) = N(\frac{1}{\sigma_n^2} A^{-1}\mathbf{X}\mathbf{y}, A^{-1}) \quad (8)$$

where $A = \frac{1}{\sigma_n^2}\mathbf{X}\mathbf{X}^T + \Sigma_p^{-1}$. In order to make a prediction, given a test input $\tilde{\mathbf{x}}$, we compute the predictive distribution by averaging over the weights $\mathbf{w}$ and obtain:

$$P(f(\tilde{\mathbf{x}})|\tilde{\mathbf{x}}, \mathbf{X}, \mathbf{y}) = N(\frac{1}{\sigma_n^2}\tilde{\mathbf{x}}^T A^{-1}\mathbf{X}\mathbf{y}, \tilde{\mathbf{x}}^T A^{-1}\tilde{\mathbf{x}}) \quad (9)$$

where $A = \frac{1}{\sigma_n^2}\mathbf{X}\mathbf{X}^T + \Sigma_p^{-1}$. Using the so-called 'kernel trick', we obtain Gaussian Process Regression by assuming that we have a mapping $\Phi(\mathbf{x})$ that maps the original N dimensional data into a large, possibly infinite dimensional space. Replacing the independent variable $\mathbf{x}$ with its transformed version leads to the following posterior distribution:

$$P(f(\Phi(\tilde{\mathbf{x}}))|\Phi(\tilde{\mathbf{x}}), \mathbf{X}, \mathbf{y}) = N(\frac{1}{\sigma_n^2}\Phi(\tilde{\mathbf{x}})^T A^{-1}\mathbf{X}\mathbf{y}, \Phi(\tilde{\mathbf{x}})^T A^{-1}\Phi(\tilde{\mathbf{x}})) \quad (10)$$

This implies that the posterior distribution (Equation 10) is also Gaussian, with the predicted mean $\hat{\mu}(\tilde{\mathbf{x}})$ and variance $\hat{\sigma}(\tilde{\mathbf{x}})$ for a given test input $(\tilde{\mathbf{x}})$. After substituting the value of A and doing some simple matrix manipulations the predicted mean and variance, in the feature space, can be expressed as:

$$\hat{\mu}(\tilde{\mathbf{x}}) = \Phi(\tilde{\mathbf{x}})^T\Sigma_p\Phi(\mathbf{x})[\sigma_n^2 I + \Phi(\mathbf{x})^T\Sigma_p\Phi(\mathbf{x})]^{-1}\mathbf{y} \quad (11)$$

$$\hat{\sigma}(\tilde{\mathbf{x}}) = \Phi(\tilde{\mathbf{x}})^T \Sigma_p \Phi(\tilde{\mathbf{x}}) - \Phi(\tilde{\mathbf{x}})^T \Sigma_p \Phi(\mathbf{x}) [\sigma_n^2 I + \Phi(\mathbf{x})^T \Sigma_p \Phi(\mathbf{x})]^{-1} \Phi(\mathbf{x})^T \Sigma_p \Phi(\tilde{\mathbf{x}}) \quad (12)$$

Equation 11 and 12 pose a significant computational complexity in situations where the number of data points is large. One type of complexity is regarding the amount of time required to learn the hyperparameters used while constructing the covariance function. The common approach to addressing this issue is to estimate the vector of hyperparameters and the noise term by maximizing the log-likelihood of the model parameters given the data.

In this research we are using the RBF stationary covariance function given in Equation 13:

$$C(x, x') = \theta_1 \exp\left(-\frac{1}{2}\sum_{i=1}^{m}\frac{(x_i - x_i')^2}{\sigma_l^2}\right) + \theta_2 + \theta_3 \delta_{ij} \quad (13)$$

where θ=[θ_1,θ_2,θ_3,σ_l] is the vector of hyperparameters of the covariance function. The parameters θ_1 and σ_l control the overall scale in vertical and horizontal variations respectively. Here θ_2 is the bias term and $\theta_3\delta_{ij}$ is the noise term where θ_3 is the variance of the noise and δ_{ij} is the Kronecker delta function.

During the training of the Gaussian Process, these hyperparameters are optimized based on a random sample from the input data set. Thus, the hyperparameters can be optimized with any m_c random points selected from a pool of n input points, where $(m_c \leq n)$. As m_c increases the computational complexity increases exponentially.

A common way to estimate the vector of hyperparameter θ is to maximize the log-likelihood by taking its partial derivatives with respect to θ and performing gradient-descent search. If the training covariance matrix $K = \sigma_n^2 I + \Phi(\mathbf{x})^T \Sigma_p \Phi(\mathbf{x})$, the log-likelihood function can be expressed as:

$$L(\theta) = -\frac{1}{2}\log|K| - \frac{1}{2}\mathbf{y}^T K^{-1}\mathbf{y} - \frac{1}{2}n\log(2\pi) \quad (14)$$

Now the derivative of $L(\theta)$ with respect to each hyperparameter θ_j takes the form of:

$$\frac{\partial L(\theta)}{\partial \theta_j} = -\frac{1}{2}\texttt{Trace}\left[K^{-1}\frac{\partial K}{\partial \theta_j}\right] + \frac{1}{2}\mathbf{y}^T K^{-1}\frac{\partial K}{\partial \theta_j}K^{-1}\mathbf{y} \quad (15)$$

Further details of the cost function and optimization algorithm can be obtained in the following reference [19]. The algorithm for generating iterated predictions using a Gaussian Process is given in Figure 3.

D. Gaussian Process Calculation using V-formulation

The second factor which influences the computational time is the inversion of $[\sigma_n^2 I + \Phi(\mathbf{x})^T \Sigma_p \Phi(\mathbf{x})]$ which is an $n \times n$ matrix (Equation 11). As the number of data points grows, inverting a matrix of size n of this size leads to operations with complexity $O(n^3)$ which is unfeasible due to memory and processing limitations. To handle this issue, we have adopted the Gaussian process method using V-formulation developed by Foster et. al [11] [12]. In the proposed technique, the low rank approximation of larger matrices is calculated using partial Cholesky factorization. In Equation 11, the $n \times n$ matrix $(\Phi(\mathbf{x})^T \Sigma_p \Phi(\mathbf{x}))$ can be approximated by VV^T, where the $n \times m$ matrix V is constructed by partial Cholesky decomposition (PCD). These algebraic operations can be summarized as follows,

$$[\Phi(\mathbf{x})^T \Sigma_p \Phi(\mathbf{x})]_{n\times n} \overset{PCD}{\rightarrow} V_{n\times m}.V_{m\times n}^T \quad (16)$$

$$[\Phi(\tilde{\mathbf{x}})^T \Sigma_p \Phi(\mathbf{x})]_{p\times n} \overset{PCD}{\rightarrow} V^*{}_{p\times m}.V_{m\times n}^T \quad (17)$$

Equation 11 can be rewritten as,

$$\hat{\mu}(\tilde{\mathbf{x}}) = V^* V^T [\sigma_n^2 I + VV^T]^{-1}\mathbf{y} \quad (18)$$

This result leads to Lemma 1 from [11]:

Lemma 1: $V^T[\sigma_n^2 I + VV^T]^{-1} = [\sigma_n^2 I + V^T V]^{-1}V^T$

The V formulation takes advantage of the above lemma to reconstruct both Equation 11 and 12. The details on the proof can be obtain in [11]. Using Lemma 1, one can rewrite Equation 18 as,

$$\hat{\mu}(\tilde{\mathbf{x}}) = V^*[\sigma_n^2 I + V^T V]^{-1} V^T \mathbf{y} \quad (19)$$

Equation 19 is the basis of the so-called V-formulation. Instead of directly inverting the $n\times n$ matrix $[\sigma_n^2 I + VV^T]$, the algorithm addresses the inversion of $n \times m$ matrix $[\sigma_n^2 I + V^T V]$, using the partial Cholesky factorization. For smaller *m*, the regression algorithm with V formulation is much faster and always numerically stable. Further details on the adopted approach to approximate the Gaussian process calculation can be obtained in the following literature [11] [12].

VI. Lorenz Dynamics and NH_3 Laser Data

The algorithms described in the previous section were tested on a data set from an NH_3 laser, whose intensity profile exhibits buildups followed by a collapse. The exact time instances of these collapses are unpredictable. Researchers have shown the connection of the experimentally measured electric field from NH_3 laser

Input: X (Input), t (target), C (Covariance Function) R_{max} (maximum rank), $\tilde{\mathbf{x}}$ (test input) and N_{iter} (Number of iterations).
Step 1: Randomly select a subset m of n input points to train hyperparameters (where $m < n$).
Step 2: Optimize hyperparameter vector θ.
a) Initialize θ=[θ_1,θ_2,θ_3,σ_l]
b) Compute covariance matrix C using n input points.
c) Compute low-rank approximation of covariance matrix constructed in (b).
d) Construct log-arithmetic likelihood function.
e) Maximize cost function with respect to each hyperparameter.
f) Obtain the optimized hyperparameter values.
Step 3: Construct Model.
g) Redefine low-rank approximation of covariance matrix using optimized hyperparameters.
Step 4: Make Predictions
for k := 1 to N_{iter}
h) Compute covariance matrix between test points and active set.
i) Redefine low-rank approximation of covariance matrix constructed in (g).
j) Compute and Store k^{th} predictive mean and variance.
k) Update $\tilde{\mathbf{x}}$ with k^{th} predicted mean.
end
Output: $\hat{\mu}(\tilde{\mathbf{x}})$ (Predicted mean), $\hat{\sigma}(\tilde{\mathbf{x}})$ (variance)

Fig. 3. The Iterative Gaussian Process Algorithm.

system to that of the dynamics of Lorenz model [33]. The Lorenz equation, as proposed by Hankel [14], can approximate the dynamical behavior of the optically pumped NH_3 single mode laser field. The equations for the Lorenz model can be expressed as follows:

$$\dot{u} = -\sigma(u - v) \tag{20}$$

$$\dot{v} = -u(w - r) - v \tag{21}$$

$$\dot{w} = uv - bw \tag{22}$$

This set of differential equations describes a nonlinear dynamical system. For specific values of σ, b and r, the evaluation of the state vector (u, v, w) gives rise to the famous Lorenz attractor. Figure 5 shows a typical three dimensional representation of the Lorenz attractor, numerically simulated using a 4^{th} order Runge-kutta integration scheme. The state variables and the control parameters in the Lorenz equation are closely related to the physical quantities of the laser physics. The details of these relationships and their interpretations can be found in [14]. H$\ddot{u}$bner showed that the square of the state variable (u) at each time instant is analogous to the electric field intensity field of the NH_3 laser system. The Lorenz-like chaotic patterns observed in the intensity pulsation of the laser can be theoretically generated using Lorenz model with the control parameters adjusted to $\sigma = 2$, $b = 0.25$ and $r = 15$ (Figure 4).

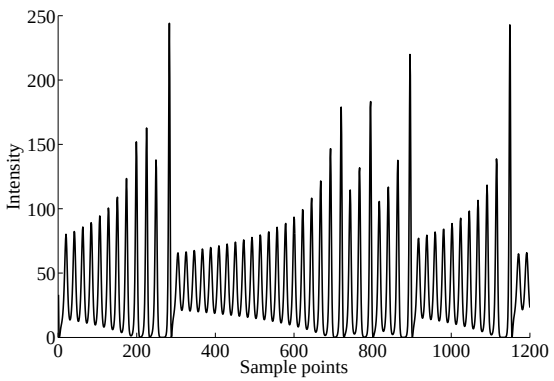

Fig. 4. This figure shows the history of the square of the state variable (u) of the Lorenz model evolved with time. The integration time step is 0.05 second with $\sigma = 2$, $b = 0.25$ and $r = 15$.

For this study, we use data from an optically pumped 81.5 micron NH_3 FIR laser. The data consists of a time series of 25,000 samples taken at a sampling rate of 12.5 MHz. The data is quantized using an 8-bit analog-to-digital converter. The data has a signal-to-noise ratio of 300. Further details of the measurement setup can be found in the literature by H$\ddot{u}$bner et al [14]. We normalize the laser intensity to lie between 0 and 1 by dividing each observed value by the maximum value of 255.

From here on, to avoid any confusion, we will label the electric field from NH_3 laser system as *experimental data set* and the u^2 obtained from Lorenz model as *simulation data set*. Given these data sets, the challenge is to build a model that can predict the behavior of the laser system as far into the future as possible with a measure of confidence.

VII. Analysis

We have tested the algorithms discussed in this paper using the data from Lorenz model and the NH_3 laser system.

A. Data Set Selection

1) Experimental Data Set: Given the $25,000$ laser intensity measurements, we divided them into two contiguous groups for training and testing as described

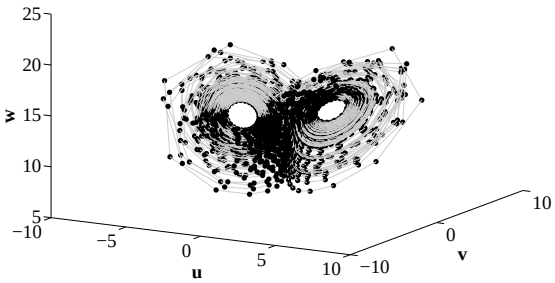

Fig. 5. This figure shows a three dimensional representation of the Lorenz attractor evolved with time. The integration time step is 0.05 second with $\sigma = 2$, $b = 0.25$ and $r = 15$. Here u, v, w are the state variables.

below. The training data was comprised of the first 1200 samples in the data set. After training we run the model against multiple test sets that were arbitrarily constructed such that the test sets did not overlap with any subset of the training data. For clarity, however, we show the results of the algorithms on three test sets only. These three cases will be referred to as test A, B, and C respectively. To find the optimal value of parameters such as the embedding dimension D and time-delay τ an exhaustive search was conducted over a specified range of discrete D and τ values. With a given training set, each combination of D and τ has been used to built the model and thereafter tested on a separate test set. For each combination, the normalized mean square error of the prediction was calculated for the same test set. Using this technique, we were able to identify the value of D and τ which minimize the normalized mean square error of the predictions. The grid search yielded $D = 35$ and $\tau = 1$. The normalized mean squared error for a set of predictions $\{\hat{y}_i\}_{i=1}^{N}$ is given by:

$$NMSE = \frac{1}{N\sigma_T^2}\sum_{i=1}^{N}(\hat{y} - y)^2 \tag{23}$$

where y_i is the observed value, $\hat{y}_i$ is the predicted value, and σ_T is the standard deviation of the true values over all the N points in the test set.

Given the training data set, we used the time series embedding methodology to generate a target vector as shown in Equation 4. Thus, the input data formed a matrix of size 34×1166 with a corresponding target vector of size 1166×1. Figure 6 shows a plot of the laser's intensity as a function of time for the training and test sets.

2) Simulation Data Set: The first 100 points of the simulated data obtained from the Lorenz model were discarded to remove transient effects. From the remaining set of data points, the first 1200 points were used for training purposes and the remaining kept for testing. Figure 4 represents the time history of the training set used in this analysis. For the simulated data the optimal value of D and τ was found to be 140 and 1 respectively, using the same grid search approach described above. Thus, the input data formed a matrix of size 139×1061 with a corresponding target vector of size 1061×1. The simulation test set was constructed beginning at time step 1201 and will be referred as test S for the rest of the paper.

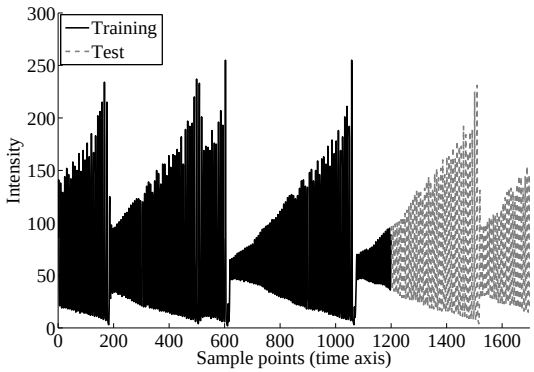

Fig. 6. This figure shows the intensity of the NH_3 laser as a function of time for the training and test sets. The objective of this paper is to create algorithms that use the first 1200 points of training data (dark color) to predict the collapses in the laser intensity in the remaining 450 points while providing a measure of confidence in the prediction. That measure of confidence will be used as a prognostic signal.

B. Results

We tested each four algorithms on these data sets: k-nearest neighbors (k-NN), bagged multilayer perceptrons (B-MLP), Gaussian Process Regression (GP), and Gaussian Process Regression developed with the V-Formulation (GP-V). We arbitrarily chose $k = 30$ for the k-nearest neighbors algorithm and averaged the output of 10 MLPs each with 10 hidden units for the bagged MLP algorithm. For the GP method, all 1166 (experiment) and 1061 (simulation) sample points of the input data matrix were used to compute the covariance matrix. We used a random sample of 50% of the input data to learn the model hyperparameters as shown in Equation 13 for the GP and GP-V algorithms. The gradient descent to find the optimal set of hyperparamters θ was done using a scaled conjugate gradient algorithm and with a maximum of 10 function evaluations.

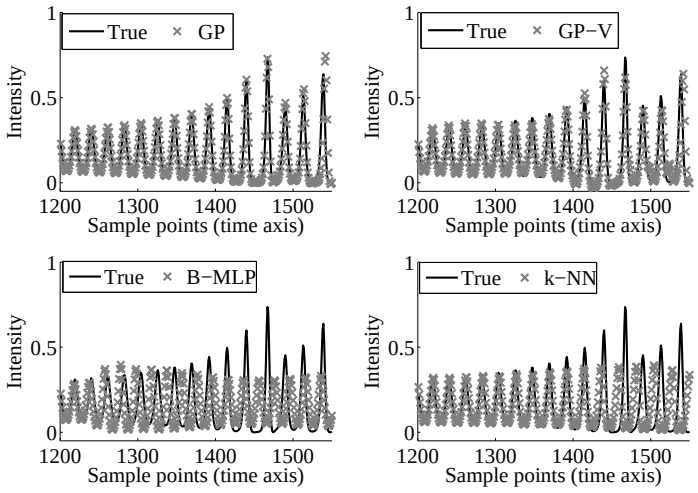

Fig. 7. The four plots of this figure show the iterative predictions (of four algorithms) overlapped on the true test data as a function of time. Both GP and GP-V algorithms are both able to model the local high frequency oscillations and the collapse event accurately. This is the same test case corresponding to the simulation data.

The GP-V algorithm has one additional parameter compared to the standard GP algorithm: we must choose the optimal rank of the V matrix. In this study, the rank was chosen such that it minimized the NMSE on a hold-out data set that did not overlap with the test set. Figure 8 shows the plot of the NMSE (average of 10 runs) as a function of rank for the NH_3 laser data set and this analysis was performed for different percentage of input data points that has been used to optimize the hyperparameters of the covariance function (Equation 13). For both simulation and experimental data sets, the optimal ranks were estimated as 10 times the corresponding dimension $(D-1)$ of their input data matrix. For each test, the algorithms are initialized with $(D-1)$ past values of the given test set, and then we generate iterated predictions as described in Section V.

TABLE I

THE TABLE SHOWS THE NORMALIZED MEAN SQUARED ERROR (NMSE) FOR THE GAUSSIAN PROCESS (GP), GAUSSIAN PROCESS WITH V FORMULATION (GP-V), THE BAGGED NEURAL NETWORK (B-MLP) AND THE k-NEAREST NEIGHBOR ALGORITHMS FOR FOUR DIFFERENT TEST SETS (A, B, C AND S). QUANTITIES IN BOLD DENOTE THE MINIMUM OBSERVED NMSE.

Test	GP	GP-V	B-MLP	k-NN
A (1201-1500)	0.1669	**0.0474**	1.5675	1.1996
B (2001-2220)	0.4873	**0.1155**	1.0431	1.2234
C (3201-3570)	**0.2377**	0.8747	1.1001	1.0767
S (1201-1550)	**0.1622**	0.1798	1.4627	1.2219

We compared the forecasts of the four algorithms described in this paper (GP, GP-V, B-MLP, k-NN) using the four test sets (A, B, C and S) mentioned earlier. The results, shown in Table I indicate that the Gaussian Process methods (with or without the low-rank approximation) produce superior results compared to the other algorithms. For test cases A and B, Gaussian process with V-formulation (GP-V) emerges with a better score compared to standard Gaussian Process (GP). In Figure 7, the true values overlapped with the predicted values (represented by $\times$) of the four algorithms that have been used for test S. Both methods modeled the local

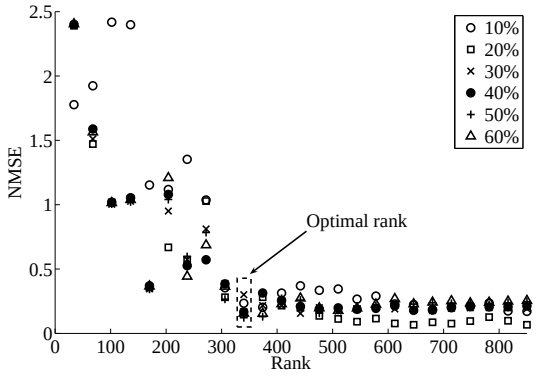

Fig. 8. This figure shows the normalized mean squared error (NMSE) of the prediction as a function of the rank of the low-dimensional approximation to the kernel matrix used in Gaussian Process Regression. The analysis was run for different percentage of input sample points that have been used to optimize the hyperparameters of the covariance function.

(rapid) oscillations as well as the global trend in the data. These methods also provided superior predictions regarding the timing of the collapse in the laser intensity when compared to k-nearest neighbor (k-NN) and bagged multilayer perceptron (B-MLP). Figure 9 shows the boxplot generated from NMSE values of the test set (of experimental data) for all the algorithms trained over 100 randomly chosen data sets. The NMSE has been calculated on a separate test set that did not overlap with any subset of the data used for training. From Figure 9 it can be concluded that standard Gaussian process (GP) and Gaussian process with V-formulation (GPV) completely outperforms k-nearest neighbor (k-NN) and bagged multilayer perceptron (B-MLP) on prediction task. This is because the medians of the NMSE of predictions for both standard Gaussian process and Gaussian process with V-formulation are much smaller compared to k-nearest neighbor and bagged multilayer perceptron. Also it can be seen that the performances of standard Gaussian process and Gaussian process with V-formulation are very comparable.

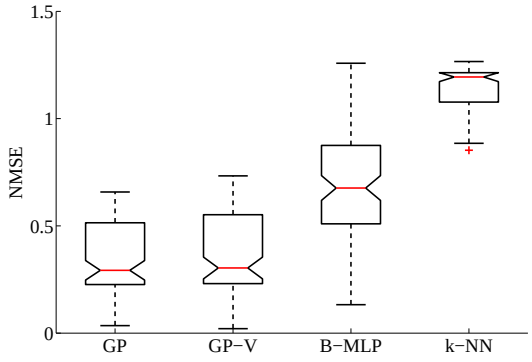

Fig. 9. This figure shows the boxplot representation of the test set using all the four algorithms. The NMSE is calculated based on 300 sample points of the test set.

To compare the rate at which prediction errors accumulate, we computed the cumulative prediction error using the following formula:

$$C_T(k) = \sum_{i=1}^{k} (\hat{y}_i - y_i)^2 \tag{24}$$

where the index k can hold any value from 1 to N and N is the total data points in the test set. In this formula, C_T represents the total normalized squared error until time step T. Using this measure, the comparative performance of the four tested algorithms yielded an interesting result. Both the Gaussian Process methods (GP and GP-V) showed significantly better performance than the bagged MLP and the k-nearest neighbor approaches. Figure 10 shows the accumulated error as a function of time for Test Set A. This plot also indicates that the GP-V algorithm outperforms the GP algorithm as indicated by the slower rate of growth in the cumulative error after the collapse. In the post-collapse region, the cumulative error drastically increases. The occurrence of a collapse event results in the loss of the GP and GP-V's prediction capability.

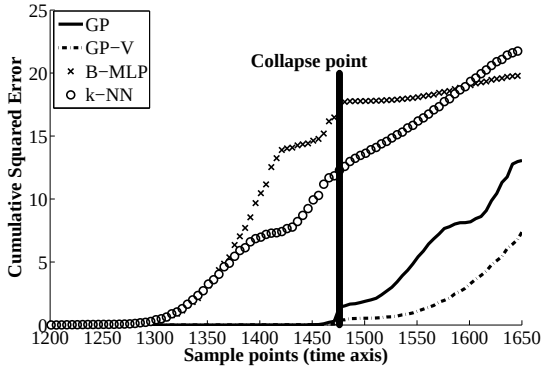

Fig. 10. This figure shows the cumulative error C_T as a function of time for the four tested algorithms for the test data shown in Figure 6. The Gaussian Process and the Gaussian Process with V-Formulation outperform the other models in terms of the cumulative error. The GP-V algorithm has the best performance out of all the models tested. The vertical line shows the time of the collapse in the laser intensity.

Another measure which we used to determine the performance of the algorithm is based on the estimated prediction horizon which is the amount of time that it takes for the cumulative error to exceed a certain (arbitrary) threshold. The estimated prediction horizon (in sample points) is shown in Table II for each of the tested algorithms. The GP and GP-V methods had the best prediction horizon, with the GP-V algorithm showing the best overall performance across the four test sets.

The threshold to determine the prediction horizon is dependent on the prognostic application. Since the cumulative error is a monotonically increasing function of time, the threshold will only be exceeded once.

Figure 11 shows the GP-V predictions for test data set A along with the associated prediction variance. This variance signal can be used for prognostics because it is generated in addition to the predictions of the time series. The figure shows that the GP-V algorithm can model both the local variations in the data along with the collapse events.

The prognostic signal is developed by placing a

TABLE II

THIS TABLE SHOWS THE TIME (IN SAMPLE POINTS) AT WHICH THE CUMULATIVE ERROR C_T EXCEEDS THE ARBITRARY THRESHOLD OF UNITY FOR EACH OF THE FOUR MODELS TESTED ON THE FOUR TEST DATA SETS. THIS NUMBER IS INDICATIVE OF THE MODEL'S PREDICTION HORIZON. THE GAUSSIAN PROCESS WITH V-FORMULATION SHOWS THE BEST PERFORMANCE WITH THE LONGEST PREDICTION HORIZON. THE NUMBERS IN BOLD INDICATE THE BEST PERFORMING ALGORITHM FOR EACH TEST SET.

Test	GP	GP-V	B-MLP	k-NN
A (1201-1650)	274	**346**	122	119
B (2001-2450)	184	**209**	144	115
C (3201-3650)	312	**313**	235	160
S (1201-1550)	**342**	341	94	241

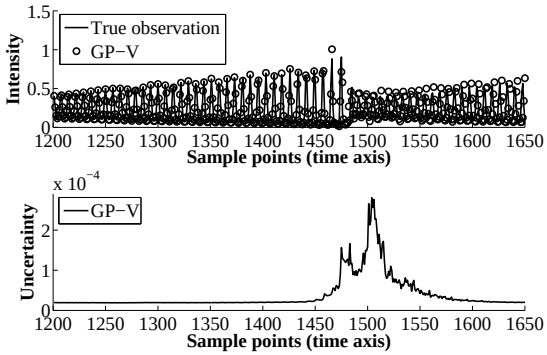

Fig. 11. The top panel of this figure shows the iterated prediction of the GP-V model as a function of time and the bottom panel shows the uncertainty in the prediction as estimated by the Gaussian Process. Notice that the uncertainty in the prediction increases dramatically just before the collapse. A thresholded version of this signal is used for prognostics.

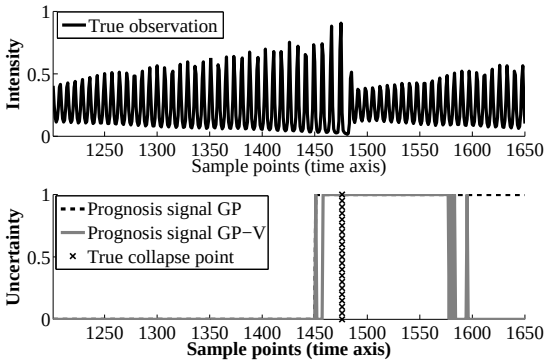

Fig. 12. The top panel of this figure shows the observed test data as a function of time. The lower panel shows a prognostic signal which is developed by placing a threshold on the uncertainty of the predictions. The GP and GP-V algorithms are both able to indicate the collapse before the event actually occurs. This is the same test case illustrated in Figure 6.

threshold on the estimated variance. When that variance crosses the threshold it is used to indicate that a collapse will occur in the near future. The threshold value can be used to control the false-positive and true-positive rates. Figure 12 shows the true observations for test set A along with the prognostic signal. The lower panel of the figure shows the prognostic signal generated by the GP and GP-V algorithms for the test data set A along with the position of the true point of collapse. Note that the prognostic signal is set to high, ahead of the actual collapse event. Thus, the algorithm's assessment of its uncertainty increases as the collapse approaches. Once the estimated uncertainty crosses a preset threshold, the output is set to "high"' status, indicating a probable collapse might occur.

Table III shows the number of sample points by which the prognosis signal leads the actual collapse point for the four test sets. For test set A, both GP and GP-V flags the warning at the same time instance but this may vary for other test cases as demonstrated in Table III.

TABLE III

THE VALUES IN THE FOLLOWING INDICATES THE NUMBER OF SAMPLE POINTS BY WHICH THE PROGNOSTIC SIGNAL LEADS THE TRUE COLLAPSE POINT. THE PROGNOSIS SIGNAL IS SET TO UNITY ONCE THE UNCERTAINTY ASSOCIATED WITH EACH PREDICTION CROSSES A PREDEFINED THRESHOLD.

Test	GP	GP-V
A (1201-1650)	24	24
B (2001-2450)	30	8
C (3201-3650)	34	42
S (1201-1550)	175	167

In spite of the superior performance over other existing methods, the applicability of standard Gaussian process (GP) for making predictions using large data sets is limited by the computational complexity of the algorithm.

The V-formulation, developed by Foster et. al. can resolve the computational issues of the algorithm. Figure 13 represents the computational time involved in training each algorithm with varying sizes of (training) data sets. The k-NN algorithm shows to be the fastest since it requires a match of one vector against the library of N data points. However, in most test cases, k-NN has the worst performance in prediction as shown in Table I and Figure 9. Figure 13 also demonstrates the superior capability of GP-V in handling both time and memory requirements with increasingly large training sets. For the laser data, the GP-V formulation shows more than a 50% reduction in computation time for a data set more than an order of magnitude larger. The standard

GP is unable to handle the memory requirement when the number of training points exceeds a certain range as indicated in Figure 13. Figure 13 shows the use of 23000 sample points for training and the remaining 2000 points for testing. On other data sets, our studies indicate that GP-V can perform well on data sets with more than a million points.

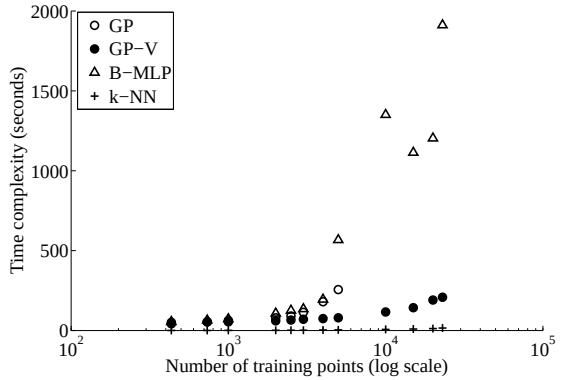

Fig. 13. This figure shows a comparison between the standard GP algorithm, the GP-V algorithm, the B-MLP and the k-NN in terms of computational time. Though k-NN shows the least time complexity but very often it has the worst performance in prediction as shown in Table I and Figure 9. The GP-V shows superior performance with respect to computational time, particularly for larger data sets. Note that the x-axis is logarithmic.

VIII. Conclusions

This paper has discussed methods to generate prognostic signals for dynamical systems that can be described using low dimensional differential equations. The dynamical system studied exhibits low-dimensional chaos and provides a good framework for studying prognostic algorithms. The paper demonstrates the use of two standard algorithms for time series prediction: the k-nearest neighbor and neural network approach. We also discuss the novel application Gaussian Process Regression to prognostics and demonstrate superior performance. Furthermore, we describe the formulation and application of a novel Gaussian Process Regression method recently developed by other researchers and apply it to the prognostics problem and provide a comparative study of the performance of the various approaches.

The methods we have developed can accurately predict the high frequency components of the signal along with the trend of the signal. We show a novel method based on analyzing the cumulative error signal to compute the prediction horizon for a prognostic model. This prediction horizon is tunable with a single parameter and provides a consistent framework for comparing predictive algorithms.

The predictions made by the algorithms shown here generally correlate well with the original signal, with the exception of the region near the collapse of the signal. Near the region of collapse, the Gaussian Process algorithm supersedes the other algorithms in terms of prediction accuracy and length of the prediction horizon.

The Gaussian Process method provides a principled approach to modeling uncertainty in predictions. This "error bar" is generated by the algorithm and is used as a prognostic signal. These signals can be used to test prognostic capabilities for a variety of dynamical systems.

References

[1] M. Akino, T. Mihara and K. Yamanaka, *Fatigue crack closure analysis using nonlinear ultrasound*, vol. 700, Feb 2004, Proc. Quantitative Nondestructive Evaluation AIP Conference, 1256–1263.

[2] A. E. Badel, D.Guegan, L. Mercier and O. Michel, *Comparison of several methods to predict chaotic time series*, Acoustics, Speech, and Signal Processing, IEEE **5** (April 1997), 3793–3796.

[3] L. Breiman, *Bagging predictors*, Machine Learning **24** (1996), no. 2, 123–140.

[4] T. Buzug and G. Pfister, *Optimal delay time and embedding dimension for delay-time coordinates by analysis of the global static and local dynamical behavior of strange attractors*, Phys. Rev. A **45** (1992), 7073–7084.

[5] F. K. Chang, *Built-in damage diagnostics for composite structures*, vol. 5, Aug 1995, Tenth International Conference on Composite structures (ICCM-10), 283–289.

[6] P. Cvitanovic, R. Artuso, R. Mainieri, G. Tanner, G. Vattay, N. Whelan and A. Wirzba, *Chaos: Classical and quantum*, http://ChaosBook.org, 2007.

[7] M. Dong and D. He, *Hidden semi-markov model-based methodology for multi-sensor equipment health diagnosis and prognosis*, European Journal of Operational Research **178** (2007), 858–878.

[8] M. Dong and D. He, *A segmental hidden semi-markov model based diagnostics and prognostics framework and methodology*, Mechanical Systems and Signal Processing **21** (2007), 2248–2266.

[9] D. Draper, *Assessment and propagation of model uncertainty*, Journal of the Royal Statistical Society, Series B (Methodological) **57** (1995), 45–97.

[10] C. H. Foong, E. Pavlovskaia, M. Wiercigroch and W. F. Deans, *Chaos caused by fatigue crack growth*, Chaos, Solitons and Fractals **16** (2003), 651–659.

[11] L. Foster, A. Waagen, N. Aijaz, M. Hurley, A. Luis, J. Rinsky, C. Satyavolu, P. Gazis, A. N. Srivastava and M. Way, *Improved linear algebra methods for redshift computation from limited spectrum data*-II, (2008), Technical Report: NASA/TM-2008-214571, Document ID: 20080008853.

[12] L. Foster, A. Waagen, N. Aijaz, M. Hurley, A. Luis, J. Rinsky, C. Satyavolu, M. Way, P. Gazis and A. Srivastava, *Stable and efficient gaussian process calculations*, Journal of Machine Learning Research (2008, submitted).

[13] T. Hastie, R. Tibshirani and J. Friedman, *The elements of statistical learning: Data mining, inference, and prediction*, Springer, 2001.

[14] U. Hübner, C. O. Weiss, N. B. Abraham and D. Tang, *Lorenz-like chaos in* NH3-FIR *lasers (data set a)*, Time Series Prediction:Forecasting the Future and Understanding the Past (N. Gershenfeld A. Weigend, ed.), 1994.

[15] J.D.Wichard and M. Ogorzalek, *Iterated time series prediction with ensemble models*, Proceedings of the 23rd IASTED International Conference on Modeling, Identification and Control, 2004.

[16] J. Juang, *Applied system identification*, Prentice-Hall, 1994.

[17] E. Keogh, K. Chakrabarti, M. Pazzani and S. Mehrotra, *Dimensionality reduction for fast similarity search in large time series databases*, Knowledge and Information Systems **3** (2001), no. 3, 263–286.

[18] J. Lin, E. Keogh, S. Lonardi, and B. Chiu, *A symbolic representation of time series, with implications for streaming algorithms*, June 2003, Proc.of 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, 2–11.

[19] D.J.C. MacKay, *Gaussian processes - a replacement for supervised neural networks?*, Advances in Neural Information Processing Systems **9** (1997).

[20] J. McNames, *Local averaging optimization for chaotic time series prediction*, Neurocomputing **48** (Oct. 2002), 279–297.

[21] R. M. Neal, *Bayesian learning for neural networks*, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1996.

[22] L. R. Rabiner, *A tutorial on hidden markov models and selected applications in speech recognition*, Proceedings of the IEEE **77** (1989), no. 2, 257–286.

[23] C. E. Rasmussen, *Evaluation of gaussian processes and other methods for non-linear regression*, 1996, Ph.D. thesis,Department of Computer Science, University of Toronto.

[24] C. E. Rasmussen and C.K.I Williams, *Gaussian processes for machine learning*, MIT Press, 2006.

[25] T. Sauer, *Time series prediction by using delay coordinate embedding*, Time Series Prediction: Forecasting the Future and Understanding the Past (N. Gershenfeld A. Weigend, ed.), 1994.

[26] M. Seeger, *Gaussian processes for machine learning*, International Journal of Neural Systems **14** (2004), 1–38.

[27] A. N. Srivastava, *Discovering system health anomalies using data mining techniques*, Proceedings of the 2005 Joint Army Navy NASA Airforce Conference on Propulsion (2005).

[28] F. Takens, *Detecting strange attractors in turbulence*, Lecture Notes in Mathematics (L. S. Young D. A.Rand, ed.), vol. 898/1981, Springer Berlin / Heidelberg, 2007, 366–381.

[29] I. A. Viktorov, Rayleigh and Lamb Waves: Physical Theory and Applications, Plenum Press, New York, 1967.

[30] E. Wan, *Time series prediction by using a connectionist network with internal delay lines*, Time Series Prediction: Forecasting the Future and Understanding the Past (N. Gershenfeld A. Weigend, ed.), 1994.

[31] M. J. Way and A. N. Srivastava, *Novel methods for predicting photometric redshifts from broad band photometry using virtual sensors*, The Astrophysical Journal **647** (2006), 102–115.

[32] A. Weigend and N. Gershenfeld, *Time series prediction: Forecasting the future and understanding the past*, Addison-Wesley, 1994.

[33] C. O. Weiss, W. Klische, N. B. Abraham and U. Hübner, *Comparison of* NH3 *laser dynamics with the extended lorenz model*, Applied Physics B **49** (1989), 211–215.

[34] M. Wenger, P. Blanas, R. J. Shuford and D. K. Das-Gupta, *Acoustic emission signal detection by ceramic/polymer composite piezoelectrets embedded in glass-epoxy laminates*, Polymer Engineering and Science **36** (1996), 2945–2954.

[35] D. C. Worlton, *Ultrasonic testing with lamb waves*, Nondestructive Testing **15** (1957), 218–222.

[36] K. Yamanaka, T. Mihara and T. Toshihiro, *Evaluation of closed cracks by analysis of subharmonic ultrasound*, Non-Destructive Testing and Condition Monitoring **46** (Nov 2004), 666–670.

[37] K. Yamanaka, T. Mihara and T. Tsuji, *Evaluation of closed cracks by model analysis of subharmonic ultrasound*, Japanese journal of applied physics **43** (Sep 2004), 3082–3087.

[38] G. U. Yule, *On a method of investigating periodicities in disturbed series, with special reference to wolfer's sunspot numbers*, Philosophical Transactions of the Royal Society of London **226** (1927), 267–298.

[39] B. Zhang and D. Cho, *Evolving neural trees for time series prediction using bayesian evolutionary algorithms*, IEEE Symposium on Combinations of Evolutionary Computation and Neural Networks, 2000.

ACKNOWLEDGMENT

This research was supported by the Integrated Vehicle Health Management Project in the NASA Aviation Safety Program, and a grant to the Arizona State University from NASA Ames Research Center. The grant was funded by the NASA Engineering Safety Center grant number NNA07CN64A, technical monitor Dr. Rodney Martin. The authors would like to thank Drs. Leslie Foster, Nikunj Oza, Kanishka Bhaduri, Paul Gazis and Ms. Elizabeth Foughty for providing important feedback while conducting this research. The authors also thank the reviewers for providing input to improve the paper.